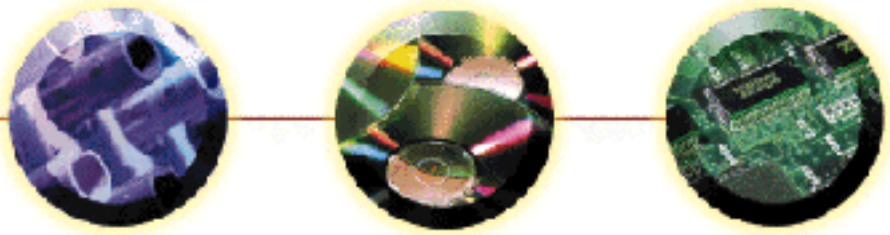
spectrum

enterprise manager

Distributed SpectroSERVER



CABLETRON SYSTEMS

Notice

Cabletron Systems reserves the right to make changes in specifications and other information contained in this document without prior notice. The reader should in all cases consult Cabletron Systems to determine whether any such changes have been made.

The hardware, firmware, or software described in this manual is subject to change without notice.

IN NO EVENT SHALL CABLETRON SYSTEMS BE LIABLE FOR ANY INCIDENTAL, INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING BUT NOT LIMITED TO LOST PROFITS) ARISING OUT OF OR RELATED TO THIS MANUAL OR THE INFORMATION CONTAINED IN IT, EVEN IF CABLETRON SYSTEMS HAS BEEN ADVISED OF, KNOWN, OR SHOULD HAVE KNOWN, THE POSSIBILITY OF SUCH DAMAGES.

Copyright © September 1998, by Cabletron Systems, Inc. All rights reserved.

Printed in the United States of America.

Order Number: 9032770 E1

Cabletron Systems, Inc.
P.O. Box 5005
Rochester, NH 03866-5005

SPECTRUM, the **SPECTRUM IMT/VNM** logo, **DCM**, **IMT**, and **VNM** are registered trademarks, and **SpectroGRAPH**, **SpectroSERVER**, **Inductive Modeling Technology**, **Device Communications Manager**, and **Virtual Network Machine** are trademarks of Cabletron Systems, Inc.

Adobe and **Acrobat** are trademarks of Adobe Systems, Inc.
C++ is a trademark of American Telephone and Telegraph, Inc.
UNIX, **OSF/1** and **Motif** are registered trademarks of The Open Group.
X Window System is a trademark of the X Consortium.
Ethernet is a trademark of Xerox Corporation.

Virus Disclaimer

Cabletron Systems makes no representations or warranties to the effect that the Licensed Software is virus-free.

Cabletron has tested its software with current virus checking technologies. However, because no anti-virus system is 100% reliable, we strongly caution you to write protect and then verify that the Licensed Software, prior to installing it, is virus-free with an anti-virus system in which you have confidence.

Restricted Rights Notice

(Applicable to licenses to the United States Government only.)

1. Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

   Cabletron Systems, Inc., 35 Industrial Way, Rochester, New Hampshire 03866-5005.

2. (a) This computer software is submitted with restricted rights. It may not be used, reproduced, or disclosed by the Government except as provided in paragraph (b) of this Notice or as otherwise expressly stated in the contract.

   (b) This computer software may be:

   (1) Used or copied for use in or with the computer or computers for which it was acquired, including use at any Government installation to which such computer or computers may be transferred;

   (2) Used or copied for use in a backup computer if any computer for which it was acquired is inoperative;

   (3) Reproduced for safekeeping (archives) or backup purposes;

   (4) Modified, adapted, or combined with other computer software, provided that the modified, combined, or adapted portions of the derivative software incorporating restricted computer software are made subject to the same restricted rights;

   (5) Disclosed to and reproduced for use by support service contractors in accordance with subparagraphs (b) (1) through (4) of this clause, provided the Government makes such disclosure or reproduction subject to these restricted rights; and

   (6) Used or copied for use in or transferred to a replacement computer.

   (c) Notwithstanding the foregoing, if this computer software is published copyrighted computer software, it is licensed to the Government, without disclosure prohibitions, with the minimum rights set forth in paragraph (b) of this clause.

   (d) Any other rights or limitations regarding the use, duplication, or disclosure of this computer software are to be expressly stated in, or incorporated in, the contract.

   (e) This Notice shall be marked on any reproduction of this computer software, in whole or in part.

Contents

Chapter 1 Overview

Why Distributed SpectroSERVER? 1-3
How to Implement Distributed SpectroSERVER 1-4
Distributed SpectroSERVER Terminology and Concepts 1-5
- Landscapes 1-5
- Landscape Map 1-6
- Modeling Catalog 1-7
- Master Catalog 1-7
- User Model 1-7

Chapter 2 Setting up a Distributed SpectroSERVER Environment

Network Partitioning 2-1
- Assigning Landscape Handles 2-2
- Changing Landscape Handles 2-3

Installing Multiple SpectroSERVERS 2-4
Designating a Master Catalog 2-5
- Database Changes that Affect the Master Catalog 2-5
  - Saving New Modeling Information to the Master Catalog 2-5
  - Copying the Master Catalog to Another Landscape 2-6

Modeling the Landscapes 2-7
Example Distributed SpectroSERVER Configuration 2-11
Processd 2-13
- Reinitializing Processd 2-13

Chapter 3 Fault Tolerance

Overview of Fault Tolerance 3-1
- Precedence 3-2
- Establishing Fault Tolerance 3-3
- Generic Configuration 3-4

Appendix A Database Partitioning

Database Partitioning A-1
- When to Partition A-2
- Database Partitioning Tools A-2

Database Partitioning Files A-3
Preparing to Partition A-3
Partitioning the Database A-4
Performing Diagnostics on Your Converted Database A-7
Attributes To Be Converted A-8

Chapter 1

Overview

This chapter provides an overview of the necessities and advantages of implementing a distributed SpectroSERVER environment.

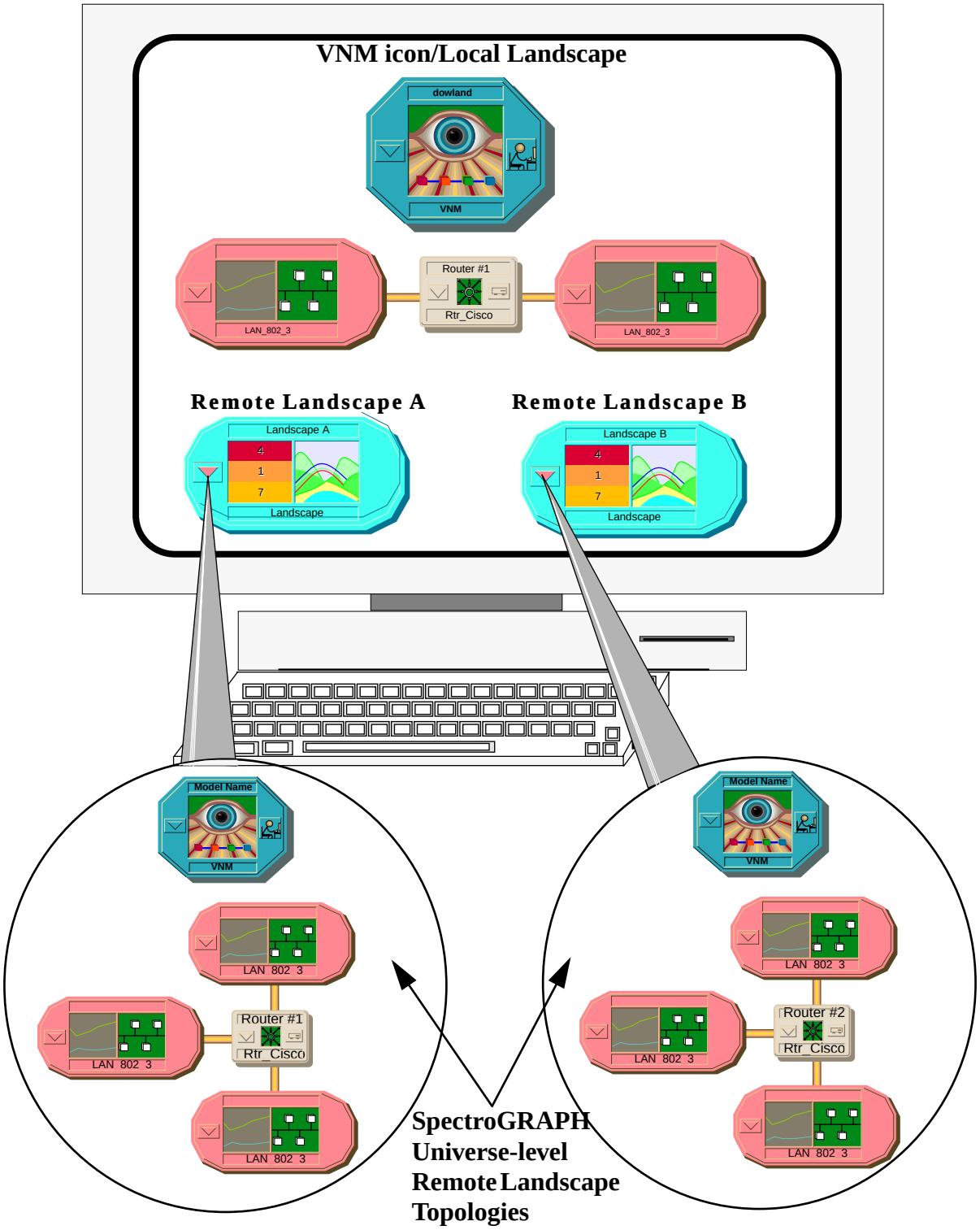
A Distributed SpectroSERVER (DSS) environment allows you to divide SPECTRUM's network management tasks among several SpectroSERVERS and provides an integrated user interface and integrated applications. When you create a network model with multiple SpectroSERVERs, it is possible for a single SpectroGRAPH or other application to access information from more than one SpectroSERVER at the same time.

DSS improves SPECTRUM performance when managing a large network by distributing the management traffic load and allowing you to delegate network management functions to remote workstations. Since the network management load is no longer concentrated on a single machine, each SpectroSERVER can be configured with a smaller load. As a result each SpectroSERVER can be administered and restarted more quickly.

The concept of **landscapes** is fundamental to DSS. Landscapes allow you to create a model of your network as a single landscape, or logically partitioned into subnets of multiple landscapes. Each SpectroSERVER can also be configured with a designated backup SpectroSERVER.

Figure 1-2 illustrates what would be seen in the user interface of a network management workstation modeling a DSS environment. This consists of a VNM model, which is representation of a SpectroSERVER, that "contains" a local landscape model and the SpectroGRAPH Universe-level network topology models. Two remote landscapes (Landscape A and Landscape B), representing two remote SPECTRUM management stations, have also been modeled in the local management station's user interface. The modeled network topologies for each remote landscape are accessed from their respective landscape models.

Figure 1-1. SpectroGRAPH Universe-level Local Landscape Topology View



Why Distributed SpectroSERVER?

Single management workstations are limited in their capacity to manage large networks. Imagine trying to manage all of the devices on the Internet with a single management workstation. When dealing with networks this large, a “divide and conquer” method is used which is just another name for distributed network management. In this type of network configuration, distributing network management is limited only by the number of machine resources dedicated to providing this management.

Many companies have sites all over the world. If a single network management station manages this type of global network, expensive wide area links would be needed. For a such a large network, management should be distributed to each of the company's sites so that most management is done locally. In this network management configuration, critical information is rolled up to a higher level management station when appropriate. This distributed setup minimizes wide area link usage, moves network management to the local level and provides the optimal network management configuration.

Distributed Management “At a Glance”
Distributed SpectroSERVER provides transparent navigation between multiple SpectroSERVERs. By simply opening a "Landscape" icon, a connection is made between the client and the SpectroSERVER which represents that "Landscape".

This allows the network manager to see information from numerous SpectroSERVERS on one screen. In addition to allowing navigation between SpectroSERVERs, the "Landscape" icon also provides an alarm summary. This can be used to determine if the remote "Landscape" should be monitored, or can safely be ignored.

Localized Polling Traffic
By using distributed SpectroSERVER, the management workstation polling the network is geographically closer to the devices it manages. This reduces traffic on wide area links and avoids congestion on the local network . Multiple "small" SpectroSERVERs generate less traffic overall than a single monolithic SpectroSERVER polling devices from a great distance.

Scalability
As a network expands, it is often less expensive to use low-end workstations as additional, dedicated SpectroSERVER machines rather than to upgrade a single workstation to a larger and faster machine to accommodate a single SpectroSERVER to managing a large non-distributed environment. This approach also avoids the issue of network expansion outpacing workstation performance improvements from year to year.

SpectroSERVER Redundancy
With Distributed SpectroSERVER, it becomes possible to set up redundancy between SpectroSERVERs. A secondary SpectroSERVER can be provided as redundant backup for a primary SpectroSERVER. Network management would continue even if the workstation running the primary SpectroSERVER fails. By simply reloading a "Landscape" database on a secondary SpectroSERVER, the network manager is protected against failures. When a failure occurs that disables the primary SpectroSERVER, the secondary automatically takes it's place and all SPECTRUM applications automatically use the redundant SpectroSERVER.

How to Implement Distributed SpectroSERVER

The network administrator needs to plan ahead as much as possible. If a network grows rapidly, a single r management workstation will become less and less responsive to network management requirements. The best way to move to a distributed network environment and avoid poor network management performance is to pro-actively plan ahead as a network needs to grow.

To distribute network management, a network administrator would look at the entire network and divide it into logical management areas. Each division would have a dedicated SpectroSERVER. Typically, a network administrator should consider dividing the network along IP boundaries or geographically. Dividing a network geographically is often the most expedient method because doing so usually follows organizational boundaries as well. This makes it easier to set up domains of responsibility and keeps the use of wide area bandwidth low since each region is responsible for managing its own portion of the network.

The goal of Distributed SpectroSERVER is to provide the user with flexible enough tools to make network management as simple as possible.

Distributed SpectroSERVER Terminology and Concepts

This section provides descriptions of the terminology and concepts used when discussing a Distributed SpectroSERVER environment.

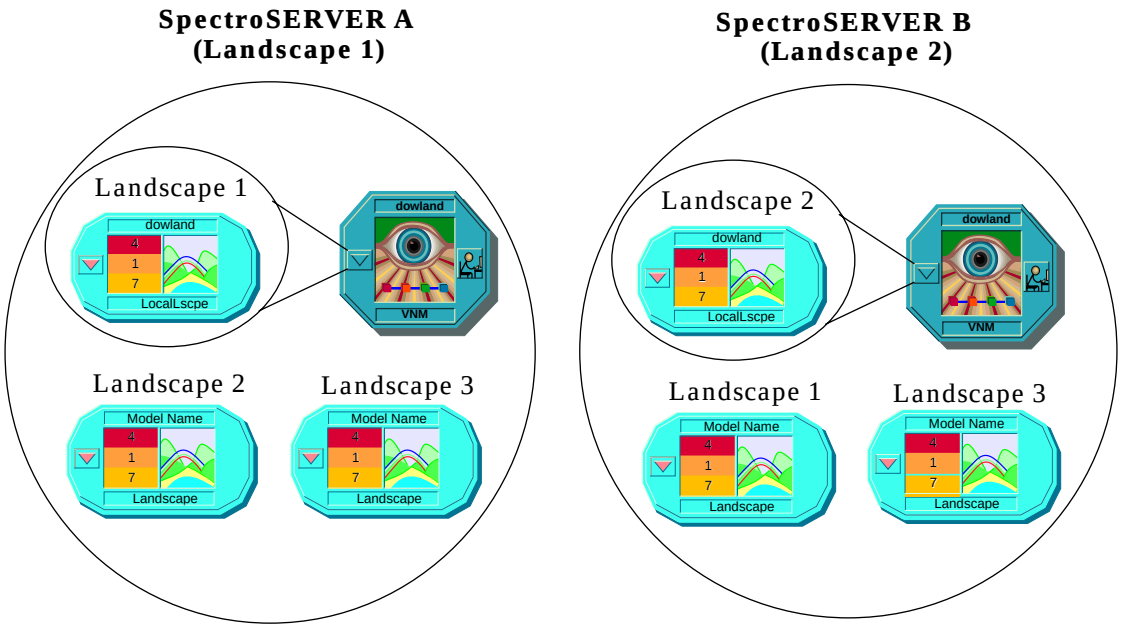
Landscapes

A landscape model is a SpectroGRAPH container model that provides a means of connecting to other SpectroSERVERs in the landscape map through the SPECTRUM user interface.

A landscape is a set of models that SPECTRUM recognizes as belonging to a particular SpectroSERVER. The landscape serves as a means of grouping these models. If desired, each landscape can be represented by a landscape icon in SpectroGRAPH. The landscape icon provides a graphical representation of a SpectroSERVER database. Through double-click zones and menu selections, the Landscape icon provides access to remote network models and also presents a rollup of alarm information for the devices modeled in those remote databases.

The terms “local” and “remote” are used to designate landscapes from the perspective of a particular SpectroSERVER. In Figure 1-2, “Landscape 1” is the local landscape for “SpectroSERVER A” while “Landscape 2” and “Landscape 3” are remote landscapes. “Landscape 2” is the local landscape for “SpectroSERVER B” while “Landscape 1” and “Landscape 3” are remote landscapes.

Figure 1-2. Local and Remote Landscapes in DSS



You can create a Landscape model at any level of the three view hierarchies; Topology, Location or Organization.

Landscape Map

SPECTRUM automatically maintains a “map” of all of the landscapes that comprise a particular DSS environment. Landscape maps contain information about the Distributed SpectroSERVERs and their databases. If any changes occur in the SpectroSERVER topology (such as a SpectroSERVER being added or removed from the network), an internal discovery mechanism updates the landscape maps of all of the other SpectroSERVERs in the DSS environment. The landscape map appears in the User Editor interface.

Modeling Catalog

A modeling catalog is a set of templates (model types, relations, etc.) installed on a particular SpectroSERVER that can be used in creating models. The set of models created through these templates make up that SpectroSERVER's landscape.

When you model multiple landscapes using Distributed SpectroSERVER, each landscape must contain identical modeling catalogs. This means that all model types that exist in one landscape's modeling catalog must also exist in the modeling catalog of every other landscape to which it is connected. If you install a new management module in one SpectroSERVER, you must install the same management module on every other SpectroSERVER in the landscape map.

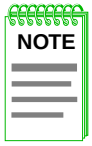
Master Catalog

The master catalog is the modeling catalog of the SpectroSERVER designated as the master. This designation is user-defined through the SpectroSERVER Control View accessed through the VNM icon. The master catalog is the SpectroSERVER that will be used to update the other SpectroSERVERs in the landscape map if any changes occur in its modeling catalog, such as installing a new management module. Creating new watches or installing new management modules must be done on the master catalog SpectroSERVER. The entire master catalog must then be manually copied to all other SpectroSERVERs in the landscape map.

User Model

A user model contains information about an individual user's permissions and other user data. This information is stored in that landscape.

Adding or modifying a user model causes the SpectroSERVER associated with that landscape to query all other SpectroSERVERs in the landscape map to see if this particular user model already exists in other landscapes. If the user model exists in other landscapes, that model is automatically updated to reflect any changes made to the initial user model. If it does not already exist, SPECTRUM adds it to all of the other landscapes.



Deleting a user model only deletes the model in that landscape. Duplicate user models on other SpectroSERVERs in the landscape map must be deleted manually.

Chapter 2

Setting up a Distributed SpectroSERVER Environment

This chapter describes procedures for setting up a distributed SPECTRUM environment. It also discusses processd, a SPECTRUM control process.

For a large network consisting of several subnets connected via wide area links and covering a large geographic area, the use of multiple landscapes using DSS is recommended. Regardless of the size of your network and the number of landscapes you create, you should consider future expansion in your network and plan your network with DSS in mind. The decision to create *multiple* landscapes depends on the size of your network and the number of wide area links connecting various subnets. For a small network, (fewer than 1,000 models) one landscape can provide a cost effective management solution.

Network Partitioning

In your network model, you can partition your network in whatever way is logical for your network application (this assumes you are modeling a network that is already configured.) However, if you are creating multiple landscapes (or plan to in the future) and you want to use **AutoDiscovery** to create your Topology hierarchy, some methods are more compatible with AutoDiscovery than others.

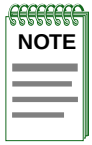
The objective is to minimize the duplication of models in your landscapes and to limit the number of connections between landscapes. Some models should be duplicated to facilitate fault isolation, such as a router that connects two IP subnets.

- Your first choice for partitioning should be to create partitions that are subnets, defined by IP address boundaries. IP address ranges provide the most convenient means of restricting AutoDiscovery exploration.

- If IP address boundaries are already established, and they do not provide a means of separating landscapes, then your second choice is to establish unique community names to differentiate devices in each landscape. This method requires more effort, since these names must be added to each device's community names table.
- Your last choice is to create partitions according to some other parameter (time-zone, state, etc.) that consists of a mix of IP subnet addresses. With this choice, you will probably have to model your network manually. However, even with this choice you should maintain as few connections between landscapes as possible and, of course, you must observe modeling rules. When too many connections exist, you begin to compromise the advantages of Distributed SpectroSERVER.

Assigning Landscape Handles

Distributed SpectroSERVER identifies each landscape by its landscape handle. Each landscape must have a unique landscape handle. You can assign any decimal number from 4 to 16,376 as a landscape handle. You can also assign a landscape handle in a hexidecimal value in the range of 0x100000 - 0xffe00000 where the lower 20 bits are set to 0. The encoded landscape handle appears at the top of all views associated with that landscape.



It is recommended that you use a sequential numbering scheme when establishing landscape handle(s). At first glance it makes sense to associate a landscape handle with a significant feature of the landscape it represents, such as a building number or some portion of a subnet IP address. However, your entry is encoded into the most significant 14 bits of the landscape handle, and the result may not appear to relate to the landscape feature you wanted.

There are two ways to assign a landscape handle:

- A landscape handle can be assigned when installing SPECTRUM. Refer to the ***SPECTRUM Installation Guide***.
- A landscape handle can be assigned by using a utility called **lh_set**.

To assign a landscape handle with the **lh_set** utility, do the following:

1. Change directories to the <*SPECTRUM Directory Path*>SS directory
2. Type the following:

```
../SS-Tools/lh_set <landscape handle>
```

3. Press <**Return**>

The new landscape handle can be specified in either decimal or hexadecimal notation. If you use decimal notation, the **lh_set** utility converts your entry into a hexidecimal landscape handle.

You must run the **lh_set** utility BEFORE you run SpectroSERVER for the first time. If you run SpectroSERVER before running **lh_set**, SPECTRUM assigns a default landscape handle that is the same whenever and wherever it is assigned by SPECTRUM. This creates the potential for duplicate landscape handles when multiple landscapes are configured. This means that these landscapes can never be accessed simultaneously from the same SpectroGRAPH or application.

Changing Landscape Handles

If SpectroSERVER has been previously started, the process for changing the landscape handle of the SpectroSERVER database involves changing the landscape handle of all models that were created when SpectroSERVER was started and updating the landscape handle specified in several resource files. The landscape handle is included in the model handle of each model in the SpectroSERVER database. Changing the landscape handle requires converting all model handles from the old landscape handle to the new one.

To determine what your current landscape handle is, do the following:

1. Change directories to the <*SPECTRUM Directory Path*>/SS directory
2. Type the following:

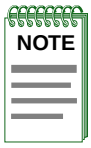
```
../SS-Tools/lh_set
```

To change the landscape handle of a SpectroSERVER database with existing models, do the following:

1. Change directories to the <*SPECTRUM Directory Path*>/SS directory
2. Type the following:

```
../SS-Tools/dbpart1/converter -db ../SS-Tools/dbpart1/
ConvTable >& converter.out
```

Your new landscape handle can be specified in either decimal or hexadecimal notation.



When you convert your landscape handle, you must also reinitialize the historical database. All historically logged information will be LOST.

Several SPECTRUM resource files specify the landscape handle. You must manually change each of these files to ensure that all SPECTRUM applications continue to function. A mismatch of landscape handles between resource files and what the SpectroSERVER is actually using can cause problems with Archive Manager, Data Export, and the Report Generator. After changing a SpectroSERVER's landscape handle, you must also change the landscape handle in the following files:

- *<SPECTRUM Directory Path>*/app-defaults/SDE
- *<SPECTRUM Directory Path>*/app-defaults/Reports

1. Change directories to the *<SPECTRUM Directory Path>*/app-defaults directory
2. Edit the app-defaults/SDE and app-defaults/Reports files and look for an entry like this:

   ```
   *PreferredLandscapes : <landscape handle>
   ```

3. Enter the value that to represents the updated landscape handle.

Installing Multiple SpectroSERVERS

When you create a distributed environment, several conditions must be met when installing multiple SpectroSERVERS and modeling multiple landscapes to represent those servers:

- Each landscape must contain an identical modeling catalog containing a database's model types and relations. This provides a consistent base for SPECTRUM intelligence.
- The same User models must exist in each landscape. User models can be seen by opening the User Editor from the VNM icon.
- The number of connections between subnets modeled in different landscapes must be minimized.
- The number of the same devices that are modeled in more than one landscape must be limited.
- Each landscape is identified by a unique "landscape handle." You must assign each landscape being modeled a unique landscape handle so that SPECTRUM can distinguish it from other landscapes in the DSS environment.

Refer to the ***SPECTRUM Installation Guide*** for details on installing SPECTRUM.

Designating a Master Catalog

You can designate a dedicated SpectroSERVER to act as a master catalog, from which other VNMs (dedicated machines) can copy models. This master catalog is part of the dedicated SpectroSERVER's database.

In order to understand the relationship between the dedicated SpectroSERVER's master catalog and the modeling catalogs on remote machines, the following should be considered.

- A master modeling catalog containing all model types purchased and installed must exist on at least one dedicated SpectroSERVER. This machine must be accessible by any distributed SpectroSERVER on the network.
- The model type data in each remote SpectroSERVER's catalog must be consistent across the network; the master catalog therefore provides a consistent base for SPECTRUM intelligence.
- Remote landscapes may contain different models and relations, depending upon those model types selected from the copied master catalog.
- If you create a new model type or application on any remote SpectroSERVER, that SpectroSERVER may become your new master catalog; you must copy that model data type to each SpectroSERVER on your network in order for the modeling catalog on each server to be consistent.

Database Changes that Affect the Master Catalog

There are two situations that result in edits being saved to the master catalog. These situations are listed below.

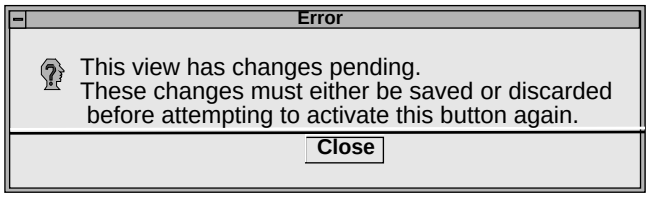
1. You create a new model type with the Model Type Editor and GenSNMPDev on your local machine. You run the mmbuild executable, and the new model type is added to model type database. You must import (copy) this new model type to every dedicated SpectroSERVER on your network that will need to establish relations with the model type.
2. You want to create a watch on one VNM. Since SpectroWATCH edits are saved to only one master catalog, the SpectroWATCH kernel determines if this VNM is the master catalog. If this VNM contains the master catalog, the edits are saved.

Saving New Modeling Information to the Master Catalog

After you create a new model application or make model type data changes with the Model Type Editor and run the mmbuild executable, you decide to use the same model type data on a remote landscape.

Before you can copy the model type data to another distributed SpectroSERVER, you must designate your current VNM as the master catalog. Follow these instructions to set this machine as the master catalog.

1. As root, start SpectroGRAPH.
2. At the Universe level, select the VNM icon, and open the Topology view.
3. Open the SpectroSERVER Control view from:
   a. the Icon Subviews menu by selecting Control
   b. the highlighted VNM icon by double-clicking the machine name label
   c. the SpectroSERVER's Performance view by clicking the Control button
4. Once the SpectroSERVER Control view opens, click Update. This updates the master catalog to reflect your recent changes.
5. The following error message may appear when you attempt to close this view.


Error
This view has changes pending.
These changes must either be saved or discarded
before attempting to activate this button again.
Close

Click Close to save the changes to the master catalog server's database.

Copying the Master Catalog to Another Landscape

You create one or more models, or create a new watch on a remote VNM with SpectroWATCH. After saving these edits to the master catalog, you now want to copy the updated master catalog to another machine. Follow the steps listed below.

1. Initially, you must add yourself as user to the dedicated SpectroSERVER with the master catalog, and to your local machine. You can do this with the User Editor. For more information, refer to the User Editor document.
2. If you are using UNIX, enter the following line at the command line:

   ```
   cd <SPECTRUM path>/SS-Tools
   ```

3. In the SS directory, enter the following executable:

   ```
   SSdbsave -catalog <catalog_file_name>
   ```

This saves the master catalog model type data to the database of the remote server. You can change the catalog file name to another meaningful string.

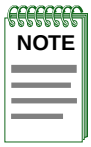
4. Copy the catalog_file_name to each remote server machine.
5. In the SS directory of the remote server, run the following executable:

```
SSdbload -catalog <catalog_file_name>
```

This loads the updated master catalog to another directory. If you want to copy the master catalog to other SpectroSERVERs, follow steps three and four.

You can check to see if the remote VNM model catalog is similar to the master catalog by either:

- using SpectroWATCH to create a new watch on the remote VNM, or
- importing a new management module built with the MTE and mmbuild. Then copy the new management module to another VNM.

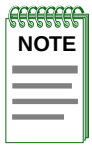


You must have a legal copy of SPECTRUM on each VNM in order for you to copy the amended model catalog from one machine to another.

Modeling the Landscapes

Several conditions must be met when modeling multiple landscapes:

- Each landscape must contain an identical modeling catalog containing a database's model types and relations. This provides a consistent base for SPECTRUM intelligence.
- The same User models must exist in each landscape.
- The number of connections between subnets modeled in different landscapes must be minimized.
- The number of the same devices that are modeled in more than one landscape must be limited.
- Each landscape is identified by a unique "landscape handle." You must assign each landscape being modeled a unique landscape handle so that SPECTRUM can distinguish it from other landscapes in the DSS environment.



It is recommended that you use a central workstation dedicated to modeling only landscapes and no other models in your Distributed SpectroSERVER setup. This will significantly reduce the workload placed on each SpectroSERVER but still allow access to all device models from this centralized, dedicated machine.

To create each Landscape model:

1. Select **Edit** from the File menu
2. Select **New Model** from the Edit menu
3. Select **Landscape** from the Select Model Type list

The Creating Landscape dialog box appears as shown in Figure 2-1.

Figure 2-1. The Creating Landscape Model Dialog

4. Enter the following information:

Model Name
The user-assigned model name. If no model name is entered, the machine name is used.

Landscape ID
The unique landscape handle that was assigned when this SpectroSERVER was installed. The landscape ID can be entered a s a decimal or hexidecimal number. To find the landscape handle for this SpectroSERVER, do the following

a. cd to the <*SPECTRUM Installation Directory*>/SS-Tools directory

b. Type *lh_set*

A message similar to the following will display with this SpectroSERVER's landscape handle:

```
Aug 26 10:27:46 ERROR at CsDbLock.cc(320): Unable to lock lock file.
SS Database landscape handle is 48 (0xc00000)
```

VNM Name
The name of the SpectroSERVER that this landscape model will represent. Generally, this is the same as the machine name on which the SpectroSERVER is installed.

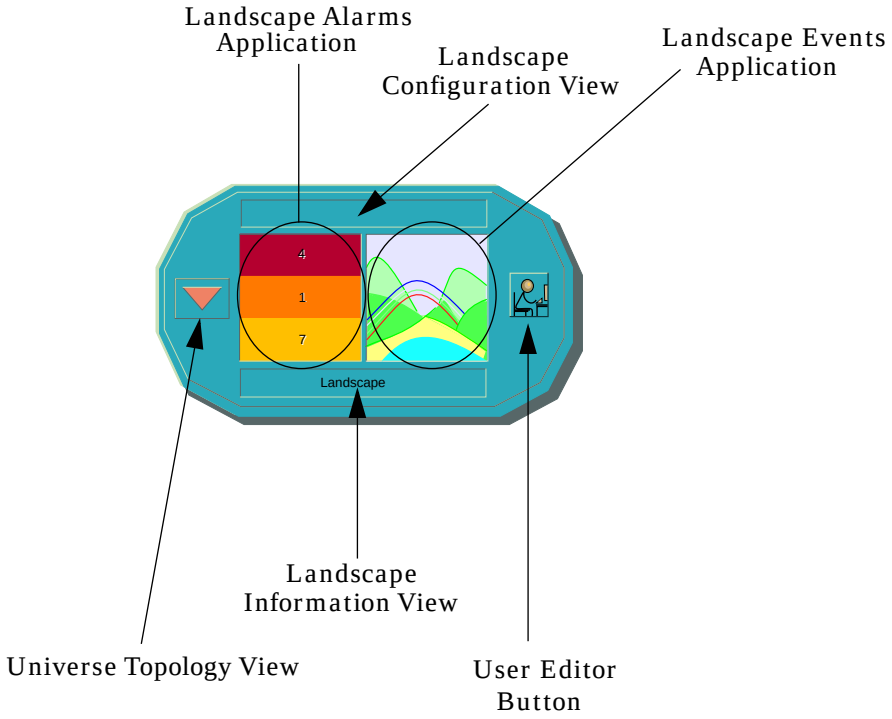
Port Number
The communication port designation as specified in the <*SPECTRUM Installation Directory*>/SS/.vnmrc file defined as comm_port=*value*. The default port number is 0xBEEF.

Security String
The desired security level for this landscape model.

5. Click **OK**

The Landscape Model appears as shown in Figure 2-2.

Figure 2-2. The Landscape Model



Example Distributed SpectroSERVER Configuration

This section provides an example of a Distributed SpectroSERVER configuration using three SpectroSERVERs as an example. In this configuration:

- SpectroSERVERs A, B, and C have each been installed with unique landscape handles (Figure 2-3).
- SpectroSERVER A is the Main Location Server and the Dedicated Master Catalog. (Figure 2-3).
- SpectroSERVER A is the centralized, dedicated workstation on which the landscape icons representing SpectroSERVER B and SpectroSERVER C are modeled (Figure 2-4).

Figure 2-3. DSS Configuration with Three SpectroSERVERs

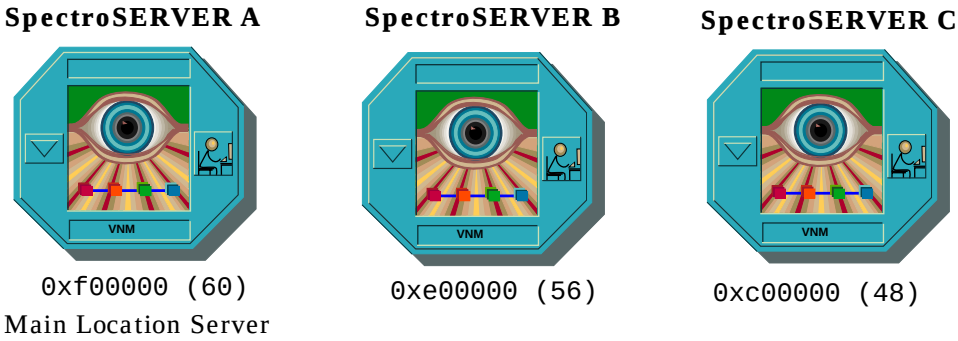
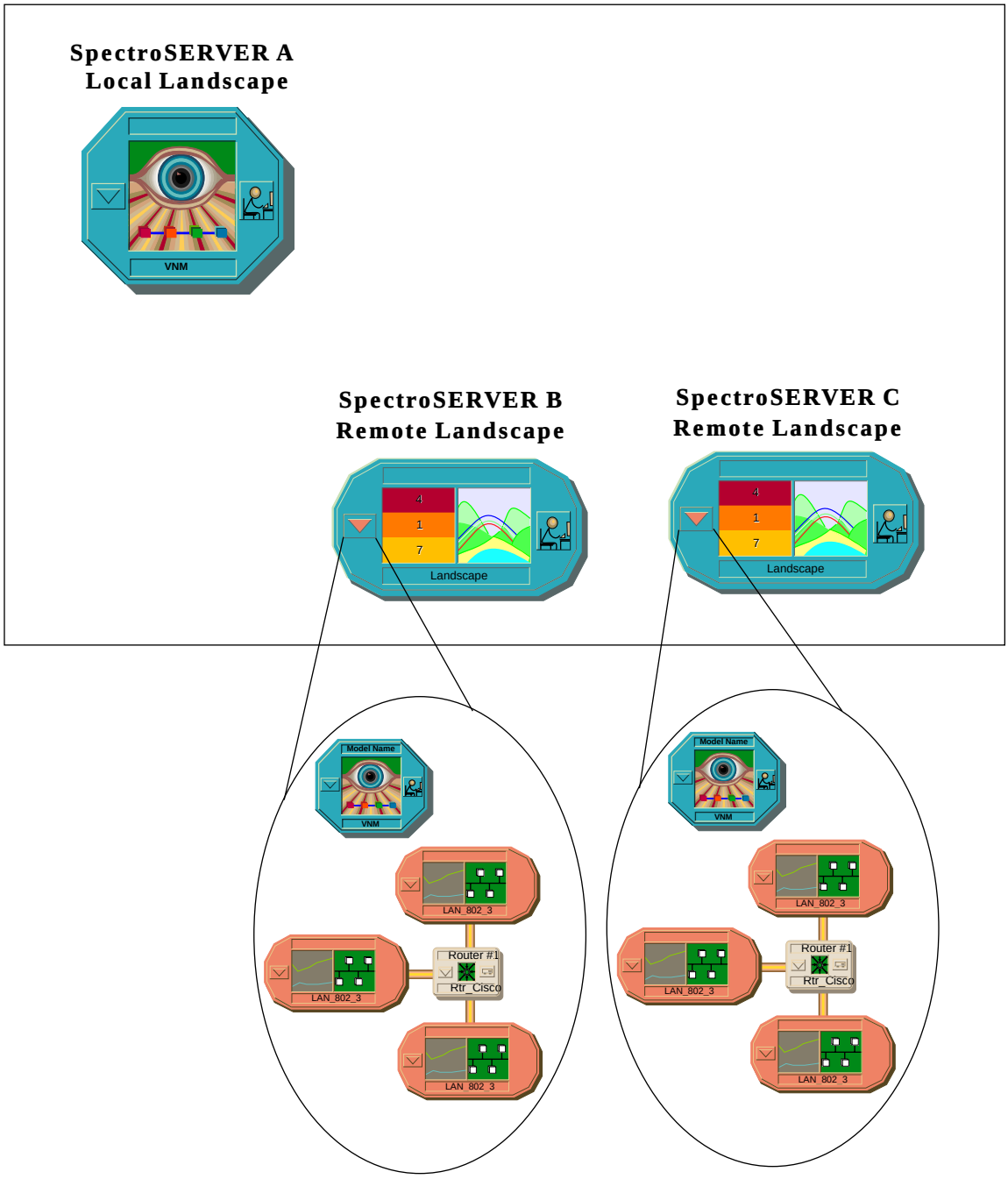


Figure 2-4. Landscape Models



Processd

Processd is a process launching and tracking daemon that provides SPECTRUM with the ability to control various processes that are run on various servers and clients in a distributed SPECTRUM environment. Processd starts processes when requested by an application such as the Control Panel. It can also start processes on system boot if configured by the user and can automatically restart critical processes in the event they die unexpectedly.

Processes are launched and tracked via processd only when an application requests such actions. Under normal circumstances, processd operates in the background and is invisible to the user. However, there are two reasons users may need to reconfigure processd data:

- Something in the processd database has become corrupted. When this occurs, the message, “Process Daemon cannot be contacted to run external application” is displayed on the screen, and processd must be stopped, reinitialized and restarted.
- A machine has been dedicated to maintain a SPECTRUM process and would like it to start system boot time using the `.autostartrc` file.

The SPECTRUM installation program makes the necessary changes so that when the system is started up the processd will also start. Processd must be run as root. This requirement is handled by system start-up as most processes executed during this time run as root.

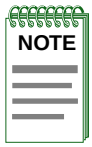
When Archive Manager is started, it is tagged with a special attribute that causes it to be restarted when it exits or is killed. This will happen even if the process is killed by a command line kill. Currently, this functionality is not user configurable. The only way to bring it down in a permanent fashion is through the Control Panel.

Reinitializing Processd

If processd fails and the message “Process Daemon cannot be contacted to run external application” is presented, it’s likely that the processd database has been corrupted and will need to be reinitialized. To reinitialize processd, do the following:

1. Become root.
2. Change directories (cd) to the SPECTRUM install area/SDPM directory. (SDPM stands for SPECTRUM Distributed Process Manager.)
3. Type “processd.sh stop” This stops processd.
4. Make sure that all SPECTRUM processes that can be shutdown are, otherwise this can cause problems with the SpectroSERVER.

5. Type “makeinstdb -dir <drive letter>:/<directory location of SPECTRUM Install Root>”.
6. Remove the CsProcdDb.k file by typing “rm CsProcdDb.k”.



Even though the database is already corrupted, removing the CsProcdDb.k file alerts the processd daemon that there is a problem and allows it to proceed with reinitialization.

Type “processd.sh start”. This will put processd back in a running state.

Chapter 3

Fault Tolerance

This chapter describes how to set up fault tolerance procedures for SpectroSERVER.

Overview of Fault Tolerance

Fault Tolerance is provided by having more than one SpectroSERVER managing a landscape. At any time, only one copy of a landscape is active. That active landscape is called the primary landscape. The other inactive copies are called secondary landscapes. When a primary landscape fails, a secondary landscape becomes active and starts managing the network. Any applications connected to the primary landscape are automatically switched to the secondary landscape. When the primary comes back, the applications are automatically switched back to the primary, and the secondary becomes inactive. Fault tolerance features can be divided into three areas:

- **switching** is the mechanism that detects the failure and switches connections to use the redundant resource.
- **readiness** measures how quickly the switch occurs to available resources, which is determined by the SpectroSERVER. A *standby*, or backup SpectroSERVER, is defined according to its different levels of readiness by temperature. That is to say that a standby is considered *hot* if it is immediately available when there is a failure. A *warm standby* is running, but may take a short time to be available. A *cold standby* is started when there is a failure.

- **data synchronization** is the measure of how closely the secondary resource tracks changes to the primary resource. It can be measured by the time delay between the change to the primary and the corresponding change to the secondary. It can also be measured by how accurately the secondary server reflects the primary server, which may vary for different kinds of data. In SPECTRUM the data is synchronized by the SpectroSERVER and database mechanisms. One of the database mechanisms is On-Line Backup, which, if configured to backup the primary server, will automatically find and backup the secondary server.

  When you model multiple landscapes, the database for each landscape must contain identical modeling catalogs. This means that all the model types that exist in one landscape's modeling catalog must also exist in the modeling catalog of every other landscape. If servers have different modeling catalogs, behavior is undefined. If you install new management modules in one landscape, you must install the same management module on every landscape.

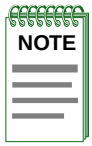
Precedence

For each landscape loaded in a VNM, the VNM/landscape map has an additional piece of information called the **precedence**, which is a number that tells SS API (SpectroSERVER Application Programming Interface) which VNM to try first to find the landscape. To ensure that two different applications do not use different VNMs at the same time when they should be using the same one, all entries in the VNM/landscape map with the same landscape handle must have a different precedences.

The servers are searched from smallest to largest precedence. The higher precedence number is for the secondary server.

The user must specify what the new precedence is and run SSdbload to enforce that the new precedence is unique. You will want to set the precedence for a machine based on its capability or network position. Refer to ***SPECTRUM Database Management*** for information on how to configure precedences.

SS API includes the intelligence needed to automatically switch an application back and forth between primary and secondary servers. SpectroGRAPH includes several features to indicate that a view is from a secondary server and adjusts the views properly. These features are active in all copies of SS API and SpectroGRAPH, but they are not used unless there is an active secondary landscape.



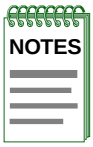
To determine what a server's precedence priority is set to you can go to the Configuration GIB View of the landscape model (particularly the local landscape model under the VNM model) or use SSdbload's -showmap option.

Establishing Fault Tolerance

Fault tolerance is intended to provide continuous service for network failures only. The secondary landscape database should not be edited except as needed to fix the network failure. Any changes made to the secondary landscape are lost when the primary database overwrites the secondary database.

To *prepare* your network for fault tolerance:

1. Designate one landscape as a primary landscape.
2. Copy the primary database to make a secondary. Use the On-Line Backup feature, or the SSdbsave utility, to make the copy of the primary landscape's database to serve as the secondary landscape. It is important that the secondary database not be a clean database.
3. Assign each SpectroSERVER a unique precedence number using `SSdbload -l -a <precedence> <savefile>` (see *SSdbload*).



The primary and secondary databases should be kept as similar as possible. The rcpd (remote copy process daemon) does this for you by working in conjunction with the On-Line backup feature to copy the databases regularly.

The secondary database will not poll devices unless it loses contact with the primary database. If you want to turn polling on in the secondary when the primary is running, add this line to the .vnmrc file: `secondary_polling=yes`. This makes the secondary server a hot standby.

To *activate* fault-tolerance:

Security: you will need to ensure that the secondary SpectroSERVER is a client to the primary SERVER. This means that client access, i.e., secured permissions, to access the backup servers must be established.

- **Start the secondary SpectroSERVER.** When this SpectroSERVER is started it becomes the secondary, due to the precedence number. Then the primary server is contacted about the secondary and the primary server forwards that information to the applications.

  Should the primary SpectroSERVER stop, applications automatically connect to the secondary server. Polling is then activated on the secondary, if it wasn't already polling. When the primary server comes back online, the applications are switched back to it. If after the switch back from the secondary server to the primary server, the secondary has `secondary_polling` set to "no," then polling is stopped.
- **Create another secondary landscape**, if you want to create a backup to the backup. With this new secondary database, if both the primary and old secondary fail, the new secondary takes over. You can create as many redundant servers as you want.

When restarting the primary server, since connections are accepted before all of the models are activated (and it may take awhile for the models to activate) then your applications will switch back to the primary server and your secondary server will turn off polling before you can use the primary server to manage your network.

To avoid such an event add this option to the command line when you restart the primary server:

`-wait_active`.

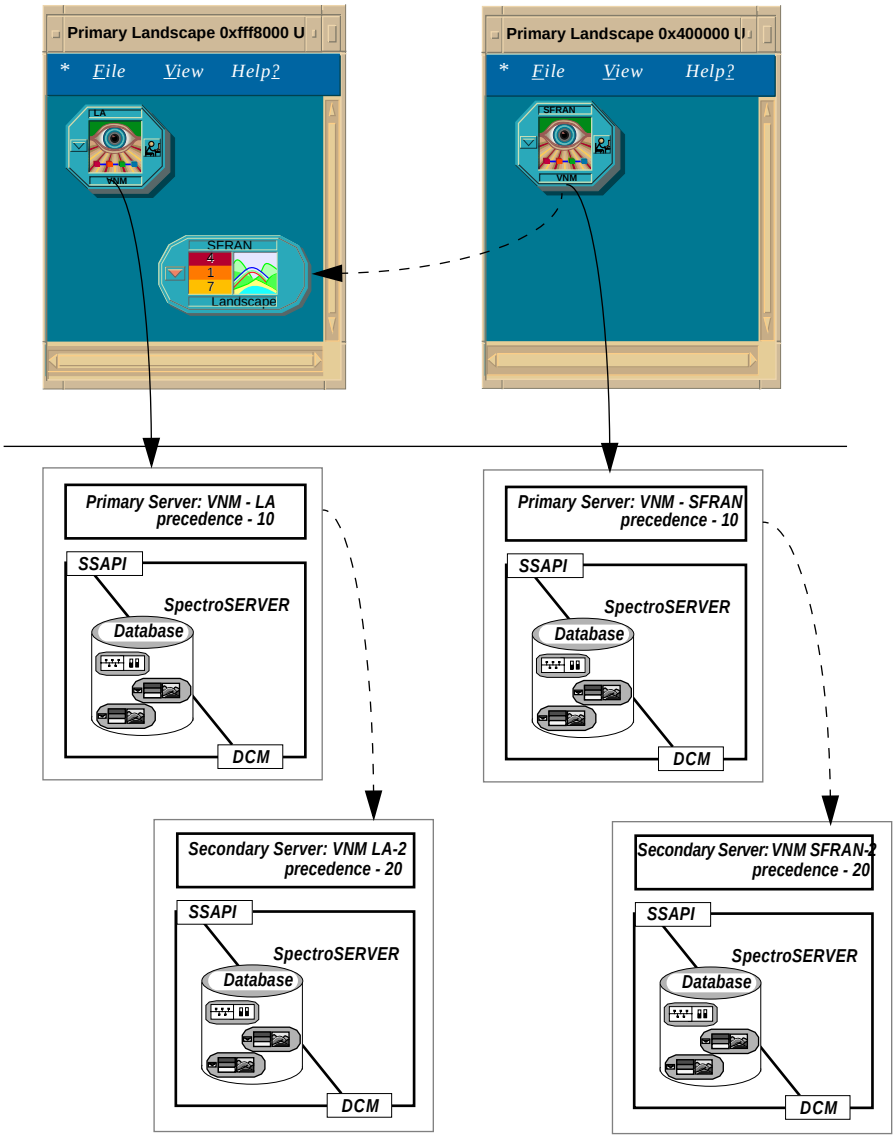
This causes the server to wait until all of the models are activated before accepting any connections. The server may appear to take longer to come up, but when it does it is completely ready to manage the network.

Generic Configuration

One possible way of configuring a network for fault tolerance is to have one secondary SpectroSERVER set up as a backup for each primary.

Figure 3-1 shows this possible network configuration. The VNM icons and landscape icon are connected via dotted lines to their corresponding databases and related precedences.

Figure 3-1. Example of Fault Tolerance



Appendix A

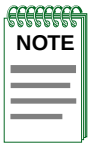
Database Partitioning

This appendix explains the concepts of, and procedures for, database partitioning SPECTRUM's SpectroSERVER.

This appendix describes partitioning procedures for the database portion of your SpectroSERVER. For more information on partitioning your network, refer to Chapter 2, *Setting up a Distributed SpectroSERVER Environment*.

Database Partitioning

This section discusses how to partition a SpectroSERVER Database, recommended practices for doing so, and utilities that assist you.



While the general procedure required for database partitioning is discussed in this chapter, the details required for assisting you with your network may not be. It is recommended that you contact Cabletron Systems' Technical Support before partitioning your database.

Partitioning the database is the process where an existing database is split into smaller portions and pieces are "restamped" with a unique landscape handle.

Restamping the database simply converts all database entities from using one landscape id to another. Association tables and model descriptors are easily found and converted. Attributes are more complicated; they come in two basic forms, fixed length data (e.g., Integer attribute), or variable length data (e.g., Octet String attribute). Fixed length attributes are easily identified and are also restamped. Variable length data attributes are difficult to convert because of their ability to store information that only its "owner" knows about. The conversion utility depends on information provided by Cabletron and 3rd party developers to determine what attributes need to be changed and how to change them. Using information from developers about their attributes allows variable length attributes to be restamped as well.

When to Partition

Partitioning preserves the work already invested in the database and should be considered if there is a significant amount of work invested in the current database (layouts, annotations, etc.). Otherwise, if the customization investment in the database is small, use AutoDiscovery to populate it. You should consider repartitioning a database only if you are dealing with a large, existing, customized SPECTRUM database and you want to change to a Distributed SpectroSERVER environment

Database Partitioning Tools

Several database partitioning tools (utility programs) are located in the `<SPECTRUM>/SS-Tools/dbpart1` directory. These tools allow you to perform partitioning tasks on your database. Table A-1 lists the tools and their capabilities.

For ease of use, add <SPECTRUM>/SS-Tools/dbpart1 to your PATH.

Table A-1. Database Partitioning Tools

Tool	Task	Access Path
lh_set	Creates a landscape handle for your database. Refer to Chapter 2, Assigning Landscape Handles.	<SPECTRUM>/SS-Tools/dbpart1/ *All of these tools accept the landscape handle in any of three forms: short decimal, long decimal and long hex.
looker	Looks at your database for landscape identifying attributes.	
converter	Converts landscape identifying attributes to use the new landscape handle.	
ConvTable	Is a collection of all the attribute information supplied by Management Module developers. In it, there is a statement about each octet attribute in the database. An attribute in the database without a corresponding entry in the ConvTable file will force the converter to stop processing the database. A missing entry usually means that a Management Module is not 4.0 compliant and hasn't supplied the install engine with the proper information needed for conversion.	

Database Partitioning Files

Several database partitioning files are created throughout the partitioning process. These files allow you to identify what is, what was, and what will be. Table A-2 lists the files and their function.

Table A-2. Database Partitioning Files

File	Task	Access Path
AUDIT.TXT	Each of the database partitioning tools produces an AUDIT.TXT file that contains debugging information about the tools operation. If the tool behaves abnormally this information is helpful in understanding why.	The file is created in the directory you are in when the tool is run.
ATTRIBUTE_HITS	The looker tool produces an ATTRIBUTE_HITS file that contains debugging information about the tools operation.	The file is created in the directory you are in when the looker tool is run.

Preparing to Partition

Partitioning a database first requires that you back up the original database, as follows:

1. Backup the original database using `SSdbsave -cm <save_file>`, or by choosing Save Database from the Control Panel's File menu.
2. Plan which models should be grouped together in which partitions. Possible ways to divide your database when partitioning are: create partitions by subnets using IP address boundaries, establish unique community names to differentiate devices, or create partitions according to some other parameter (time-zone, state, etc.) that mixes IP subnet addresses. It is **recommended** that you partition your actual network according to IP subnets.
3. Write down your current database landscape handle.
4. If you do not know the landscape handle, you can find out by running `lh_set` with no options specified, where the number displayed is the current landscape handle in the database. (See *Assigning Landscape Handles* on Page 2-2.)
5. Decide what the landscape id is going to be for each partitioned database. Remember, each converted database must have a *unique* landscape id.

Partitioning the Database

The following steps are required to partition a database.

1. Create directories for as many partitions as you are creating.

   If you want to split your database in half, you would make two directories where the first directory would contain the database that represents one half, and the second directory would contain the database that represents the other.

2. Copy the database files into each directory created.

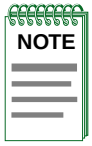
3. Along with the database files, which are the files that begin with SSDb, copy the `.vnmrc` file, the `.watchrc` file, the `SSDb.dbd` file, and any `vista.*` files into each partition directory that you created.

4. Change directories to one of the newly created directories.

5. Start the SpectroSERVER against this database. Enter the command as follows:

   ```
   ../SpectroSERVER
   ```

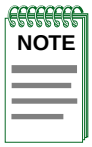
6. Edit the file used to set SPECTRUM's X defaults; `<SPECTRUM directory>/app-defaults/spectrum` file. Look for the "windowTitle" entry in this file and add the option to display the model handle in the window title bar (%H).



If you are unable to read the handle in the SpectroGRAPH window, you may want to remove one of the other entries that are displayed in the window title bar.

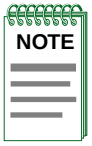
7. Then start SpectroGRAPH and connect it to the SpectroSERVER with your database loaded.

8. Follow the directions below to collect a list of the model handles of the root node of each subtree that is to be removed from this partition. You may prune by deleting a list of models or by saving a list of models and deleting the rest. For the purpose of these instructions we will be deleting rather than saving models.



Depending on your database it may be more economical for you to proceed through the pruning process by saving models instead of deleting them, e.g., there are fewer items that need to be saved than deleted. If you create a list of model handles for saving models, use the CLI script `save_models` in lieu of the CLI script `delete_models` that is used for deleting models.

a. Choose a hierarchy (Topology, Location, Org-Chart). For ease of viewing, navigate into a level that has 300 or fewer device models below it.

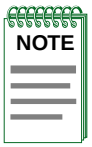


To help decide where to start, you may run **Inventory Reports** from the Universe level. Inventory Reports is available from **Generate** off of Reports in the File menu. This will give you a listing of all of the models in the hierarchy.

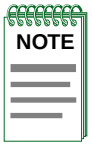
b. Now get the model handle of the root node of the first subtree you wish to remove. To do so, select a model by placing the mouse pointer on it. Then use the middle mouse button to choose a view, say Information, and read the model handle from the title bar of the view. Repeat this step for each subtree you intend to remove.

c. Repeat steps a and b for each hierarchy and level to get the rest of the subtree root nodes that belong in the list you are creating. When your list is complete you are ready to delete models from the database.

9. First shutdown SpectroGRAPH and then run the CLI command `connect` and the `delete_models` script. You must be root user to run these commands.



Add the path to SS-Tools/dbpart1 (to delete_models) and Spectrum/vnmsh (to connect) to your environmental variables PATH.

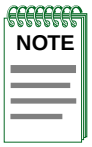


This process marks the models for deletion. SpectroSERVER actually destroys the models, so it is suggested that SpectroSERVER be allowed to run over an extended period when the server's activity is low to allow the database time to be cleaned up.

Examples of the delete-models script are shown below:

Example 1: `delete_models contains 0x400cf`

Example 2: `delete_models contains <filename>`



If you wish to remove more than one subtree from your database, use a file to specify the model handle for each subtree to be removed. The file (which contains more than one model handle) should contain one model handle per line. See Example 2 above.

10. Once the models are destroyed, shut down SpectroSERVER, disconnect from CLI and backup the pruned database using the command `SSdbsave -cm <path/save_file>`. This backup copy of the partition safeguards your work while the conversion process is taking place.

11. Repeat steps 4 through 10 for each partition directory copy of the database.

12. Before running converter, link converter to your local directory by performing the following steps at the command line in the *<SPECTRUM Installation Directory>/SS* directory:

    Step 1: `ln -s ../SS-Tools/dbpart1/converter .`

    Step 2: `ln -s ../SS-Tools/dbpart1/ConvTable .`

    To determine if the link is valid:

    For Solaris, at the command line type: `ls -alt` and check for the link to the converter executable.

    For NT, in a UNIX shell at the command line type: `ls -alt` to see if the `converter.exe` executable is present.

    Run `converter` to complete the conversion process. The conversion program restamps each database partition with a unique landscape handle, and converts each model handle in the database to use the new landscape handle. The database to be converted must be in the SS directory and this command must be made from the SS directory for this process to work. Type in:

    ```
    converter -db ConvTable <New_Landscape_id>
    ```

    This tells the program to convert the database to use the new landscape id that you specify in the command.

If there is an interruption while `converter` is running, the database will no longer be valid for continuing the conversion process. You must reload the database and start again.

A database with 5000 models takes about 2 hours to convert. To make sure that the program is running correctly, you can watch the AUDIT.TXT file by typing: `tail -f AUDIT.TXT`.

13. The conversion process is now complete for the chosen database copy. See *Performing Diagnostics on Your Converted Database* on Page A-7 to insure that the conversion was complete and accurate. Then run `SSdbsave -cm <save_file>` to save the converted database. Repeat steps 12 and 13. When the process is complete each directory will have a smaller database with a unique landscape identifier.

14. Move the converted and saved database (i.e., the `<save_file>`) along with the `.vnmrc` and `.watchrc` files to the desired platform. Expand the database on the new platform with `SSdbload -il <save_file>` and you are ready to run SPECTRUM. Repeat this step for each platform.

Performing Diagnostics on Your Converted Database

1. Bring up SpectroSERVER in debug mode to validate the conversion process. Any invalid model handle, symptomatic of an unconverted attribute, will be detected.

   a. To do this, type:

   ```
   <SPECTRUM>/SS/SpectroSERVER -debug >& SpectroSERVER.OUT
   ```

   This invokes the SpectroSERVER in debug mode and routes its output to a file. This mode of SpectroSERVER will not give you a message when it is ready.

   b. Then type `tail -f SpectroSERVER.OUT | grep -i invalid > INVALID.OUT` to search the output file for 'invalid', the keyword that is generated when an invalid model handle is used. The results are saved to INVALID.OUT.

   c. Run SpectroGRAPH against your SpectroSERVER and perform some normal operations. As you do this, monitor the window where you have the `tail` command running. If messages appear indicating an invalid model handle was used, save the list to aid in determining what went wrong.

2. Run `looker`, a program that examines the database for attributes containing landscape ids, to determine if the attributes that required change have been changed. It generates an ATTRIBUTE_HITS file for debugging purposes, and an AUDIT.TXT file. This gives you a "snap shot" of the database before and after the conversion. By comparing before and after snap shots, you can get an idea of converter effectiveness.

   a. Make sure that the SpectroSERVER is not running against the database.

   b. Run the program in the same directory as the database being scanned.

   c. Reinitialize database to the "before conversion" state.

   Run:

   ```
   <SPECTRUM>/SS-Tools/dbpart1/looker > LOOKER.BEFORE
   ```

   Proceed with conversion process. After conversion process, run:

   ```
   <SPECTRUM>/SS-Tools/dbpart1/looker
   <old_landscape_handle> > LOOKER.AFTER
   ```

The output files, LOOKER.BEFORE and LOOKER.AFTER, contain attribute values that look like the current landscape id. The looker program prints an attribute's fields as: `<name> <type> <id> <occurrences> <questionable values> <unused>`. The program also uses the following indicators to identify what looker found while examining the database.

- *Searching for...* indicates the landscape id it was looking for. This is normally 0x400000, but that might be different if the landscape was previously stamped using the lh_set program.
- *W* next to an attribute indicates a SpectroWatch attribute that can be safely ignored.
- *Used* indicates how many times an attribute of that type appeared in the database.
- *Hits* indicates the number of attribute values that could contain a model handle.
- *Empties* are attributes that don't contain any value.

Attributes To Be Converted

The following attributes have been identified in Cabletron code as needing conversion:

Icon_List	0x01002A
Button_List	0x01002B
VIB_IIB_List	0x010037
User_Shortcuts	0x010079
Segment_Connection_List	0x01007D
HU_Part_Models	0x0100B9
HU_Conn_Lst	0x010246
Collected_By_List	0x010C48
TopModelHandleList	0x011998
ServerList	0x0110C3
Repair_List	0x0114E3
User_PA_List	0x23000A
PulledBoardList	0x0D0268
Duplicate_List	0x0118E6
Pulled_Bd_List	0x010FEE
StationList	0x11cf1
PeerLinkList	0x11d3b
attr_list1	0x820000
attr_list2	0x820013
attr_list3	0x820014
attr_list4	0x820015
attr_list5	0x820016
load_hist_list	0x820004
MMACPlusIconList	0xd0268